REMARKS

Please reconsider the application in view of the following remarks. Applicant thanks the Examiner for carefully considering this application.

Disposition of Claims

Claims 1-25 are pending in this application. Claims 1, 4, 6, 16, 22, and 24 are independent. Claim 25 has been canceled by this reply. The remaining claims depend, directly or indirectly, from claims 1, 4, 6, 16, and 22.

Rejections under 35 U.S.C. § 103

Claims 1-3, 6-11, 14, 16-19, 22, and 25 stand rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 6,496,865 ("Sumsion") in view of Admitted Prior Art (APA). Independent claims 1, 6, 16, and 22 have been amended to clarify the present invention recited. Claims 2 and 25 have been canceled by this reply, thus the rejection is moot with respect to these claims. To the extent that this rejection may still apply to the amended claims, this rejection is respectfully traversed.

The present invention relates to collaboration between components in a distributed application. Specifically, the present invention restricts direct interaction between distributed collaborating components of an application by introducing an application-independent interface (*i.e.,* service layer) between distributed collaborating components and invoking a service from the application-independent interface in order to enable interaction between distributing collaborating components. This allows distributed applications to run seamlessly after the interface between the components of the application has been modified.

Further, the application-independent interface includes capability to interpret a usage specification and a logic execution specification from a client at runtime. The usage specification and/or logic execution specification are specified as arguments of the invoked service. A usage specification specifies data (object attributes) required by the client, and a logic execution specification specifies services (object methods) required by a client. Specifically, the usage specification includes the attributes of the server object needed by the client. The application-independent interface finds the server object, interprets the usage specification to

determine the selective object attributes to fetch, fetches the requested attributes from the server object, and returns the data to the client component (see, *e.g.,* page 7 of the specification).

Similarly, the logic execution specification disclosed in the present application may be specified to the application-independent interface as an argument when any business logic needs to be executed by the application-independent interface. The application-independent interface executes the logic and returns the results to the client component. The logic execution specification can be modified as necessary at runtime. Thus, if the interface of a server object changes for any reason, the logic can be modified to reflect this change. The application can continue to run without having to reset its running state because the logic execution specification is interpreted at runtime by the application-independent interface.

In contrast to the present invention, Sumsion relates to providing interpreters (*i.e.,* programs that provide a standard API to enable a single version of an application to execute on different types of operating systems) access to server resources in a distributed network (see, *e.g.,* col. 1, ll. 10-13 of Sumsion). In Sumsion, a resource access system (RAS) is installed on both the client and the server. Resource access requests generated by the interpreter application on the client node are received by the client RAS and forwarded to the server RAS. Thus, the RAS enables the interpreter system to provide server resource access independently of the type of operating system implemented in the client node.

Specifically addressing the claims of the present invention, independent claim 1 has been amended to include the limitation "wherein invoking a service from the application-independent interface comprises sending a usage specification to the application-independent interface." Applicant respectfully asserts that Sumsion fails to disclose or suggest a usage specification as defined in the present application. As noted above, a usage specification is used by the client to indicate specific *attributes* associated with objects that need to be obtained from the server by the application-independent interface. More importantly, the attributes may be associated with different objects, where the application-independent interface obtains only the specified selected attributes from each object. Therefore, the application-independent interface of the present invention does not obtain all the objects associated with a client request, but rather, only the selected attributes specified in the usage specification.

The Examiner references col. 4, ll. 60-67 of Sumsion in asserting that Sumsion discloses a usage specification. However, Applicant respectfully disagrees with the Examiner and points that that the referenced portion of Sumsion discloses only the RAS, which includes functionality to receive requests from the interpreter and translate the requests for the client redirector. However, neither the RAS nor redirector include functionality to interpret a *usage specification* from the client that specifies attributes associated with server objects that need to be obtained from the server. Further, Sumsion does not even disclose or suggest that the client specifies a usage specification to the RAS or the redirector. Thus, it is clear that Sumsion does not disclose a usage specification that is taken by the application-independent interface as an argument, interpreted by the application-independent interface, and then used to obtain selected attributes from the server in response to a client request. Moreover, it is clear that APA does not disclose or teach a usage specification.

In view of the above, Sumsion and APA, whether considered separately or in combination, fail to disclose each and every element recited in amended independent claim 1 of the present application. Thus, claim 1 is patentable over Sumsion and APA. Dependent claim 3 is patentable for at least the same reasons. Further, independent claims 6 and 16 have been amended to include similar allowable subject matter (*i.e.*, a usage specification) and are patentable over Sumsion and APA for at least the same reasons. Dependent claims 7-11, 14, and 17-19 are also patentable for at least the same reasons. Independent claim 22 recites that the service layer has capability to interpret a *specification of usage*, and is also patentable over Sumsion and APA for the reasons detailed above. Accordingly, withdrawal of this rejection is respectfully requested.

Claims 4-5, 12-13, 15, 20-21, and 23-24 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Sumsion in view of APA and further in view of U.S. Patent No. 6,760,747 ("Allard"). This rejection is respectfully traversed.

Claim 4 has been rewritten in independent form and includes the limitation "wherein invoking a service from the application-independent interface comprises sending a logic execution specification to the application-independent interface." The Examiner admits that Sumsion and APA do not teach or suggest a logic execution specification. With respect to Allard, the Examiner references col. 4, ll. 44-46 of Allard in asserting that Allard teaches a logic

execution specification. Applicant respectfully disagrees with the Examiner and points out that although Allard discloses a method that is invoked and sent to the server to add a book to an order, Allard does not disclose or suggest a logic execution specification as defined in the present application.

In the present invention, the logic execution specification is specified to the application-independent interface, and *not* directly to the server. The application-independent interface includes functionality to receive the logic execution specification as an argument, execute business logic in response to the logic execution specification, and subsequently returns the result to the client that sent the logic execution specification (*e.g.,* page 8, paragraph 27 of the present specification). The point of the present invention is to restrict direct communication between the client and the server, using the application-independent interface. In contrast, Allard teaches directly sending the method for adding a book to an order from the client to the server, without involving the logic execution specification (see, *e.g.,* col. 4, ll. 44-46 of Allard). Thus, amended claim 4, and dependent claim 5 are patentable over Sumsion, APA and Allard.

Further, amended independent claims 6 and 16, and independent claim 24, include similar allowable subject matter (*i.e.,* a logic execution specification as defined above) and are therefore patentable over Sumsion, APA, and Allard. Associated dependent claims 12-13, 15, and 20-21 are patentable for at least the same reasons.

With respect to independent claim 22, Allard also does not teach a usage specification that is used to specify selected attributes of server object to be obtained on behalf of the client by the application-independent interface (*i.e.,* the service layer). Thus, claim 22, and dependent claim 23 are patentable over Sumsion, APA, and Allard. Accordingly, withdrawal of this rejection is respectfully requested.

Conclusion

Applicant believes this reply is fully responsive to all outstanding issues and places this application in condition for allowance. If this belief is incorrect, or other issues arise, the Examiner is encouraged to contact the undersigned or his associates at the telephone number listed below. Please apply any charges not covered, or any credits, to Deposit Account 50-0591 (Reference Number 16159/023001; P6425).

Dated: December 1, 2004

Respectfully submitted,

By _____

Robert P. Lord
Registration No.: 46,479
Osha & May L.L.P.
1221 McKinney, Suite 2800
Houston, Texas 77010
(713) 228-8600
(713) 228-8778 (Fax)

78555_1.DOC